NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

ERROR ANALYSIS OF SENSOR MEASURMENTS IN A SMALL UAV

by

James Ackerman

September 2005

Thesis Advisor:	Isaac I. Kaminer
Second Reader:	Vladimir Dobrokhodov

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			Form Approved OMB No. 0704-0188
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.			
1. AGENCY USE ONLY ***(Leave blank)***	**2. REPORT DATE** September 2005	**3. REPORT TYPE AND DATES COVERED** Master's Thesis	
4. TITLE AND SUBTITLE: Error Analysis of Sensor Measurements in a Small UAV			**5. FUNDING NUMBERS**
6. AUTHOR(S) James Scott Ackerman			
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000			**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A			**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**
11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.			
12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release, distribution unlimited			**12b. DISTRIBUTION CODE**
13. ABSTRACT (maximum 200 words) This thesis focuses on evaluating the measurement errors in the gimbal system of the SUAV autonomous aircraft developed at NPS. These measurements are used by the vision based target position estimation system developed at NPS. Analysis of the errors inherent in these measurements will help direct future investment in better sensors to improve the estimation system's performance.			
14. SUBJECT TERMS Unmanned Aerial Vehicle, UAV, Target Tracking, Euler Angles, Camera Line of Sight, Minimum Function (FMINUNC), Position Estimation, Piccolo, NPS Autopilot			**15. NUMBER OF PAGES** 63
			16. PRICE CODE
17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UL

NSN 7540-01-280-5500 Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. 239-18

THIS PAGE INTENTIONALLY LEFT BLANK

Approved for Public Release; Distribution is Unlimited

ERROR ANALYSIS OF SENSOR MEASUREMENTS IN A SMALL UAV

James S. Ackerman
Ensign, United States Navy
BSEE, The Citadel, 2004

Submitted in partial fulfillment of the
Requirements for the degree of

MASTER OF SCIENCE IN ENGINEERING SCIENCE
(MECHANICAL ENGINEERING)

from the

NAVAL POSTGRADUATE SCHOOL
September 2005

Author: James S. Ackerman

Approved by: Dr. Isaac Kaminer
Thesis Advisor

Dr. Vladimir Dobrokhodov
Second Reader/Co-Advisor

Dr. Anthony Healey
Chairman, Department of Mechanical and
Astronautical Engineering

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

This thesis focuses on evaluating the measurement errors in the gimbal system of the SUAV autonomous aircraft developed at NPS. These measurements are used by the vision based target position estimation system developed at NPS. Analysis of the errors inherent in these measurements will help direct future investment in better sensors to improve the estimation system's performance.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

I would like to thank Dr. Isaac Kaminer and Dr. Vladimir Dobrokhodov for their endless patience in assisting me to accomplish this task. There were many times when I knew that the questions I was asking were trivial to them but they took the time to help me understand what I needed to do to get the job done. I would also like to thank them for their friendship that I believe we have built along the way. I thank them for giving me this challenge that will make others down the road seem simple. I will always remember them and look to the example they set of what hard work really is when I times are hard. Both of these men have done a lot with their lives and I hope I can accomplish as much some day.

I would also like to thank my wife who had to almost live without me for the months it took me to finish this thesis. The time I wasn't there for her was hard on her but she persevered through it very well and I thank her for being strong for me.

I would also like to thank Todd Trago and Tom Brashear for their help. Whether it was just being there to chat with and escape the pressure that a thesis puts on your shoulders or being there to answer my usually stupid questions I thank them. I know Todd will have a great career and I am positive Tom will continue his success.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

A. BACKGROUND

The purpose of the Small Unmanned Aerial Vehicle (SUAV) is to provide video and target position estimations to the Tactical Network Topology (TNT) Experiment being conducted by the Naval Postgraduate School. "TNT is an integrated program of quarterly field experiments that develop and demonstrate new technologies to support near term needs of the war fighter" [4]. The experiment focuses mainly on "wireless networks, autonomous vehicles, sensor networks, situational awareness and target tracking and identification" [4].

The SUAV is just one component in the TNT Experiment. The SUAV provides Vision-Based Target Tracking (VBTT) which proves detailed reconnaissance and simultaneous imagery, see Figure (1). The detailed reconnaissance includes target position estimation. Targets can be acquired using an onboard gimbaled camera. The angles of the camera (pointed toward a target) with respect to the body of the SUAV along with telemetry data from an onboard Inertial Navigation System (INS) allows us to estimate position of a target. Along with live video this target position estimation is the main mission of the SUAV.

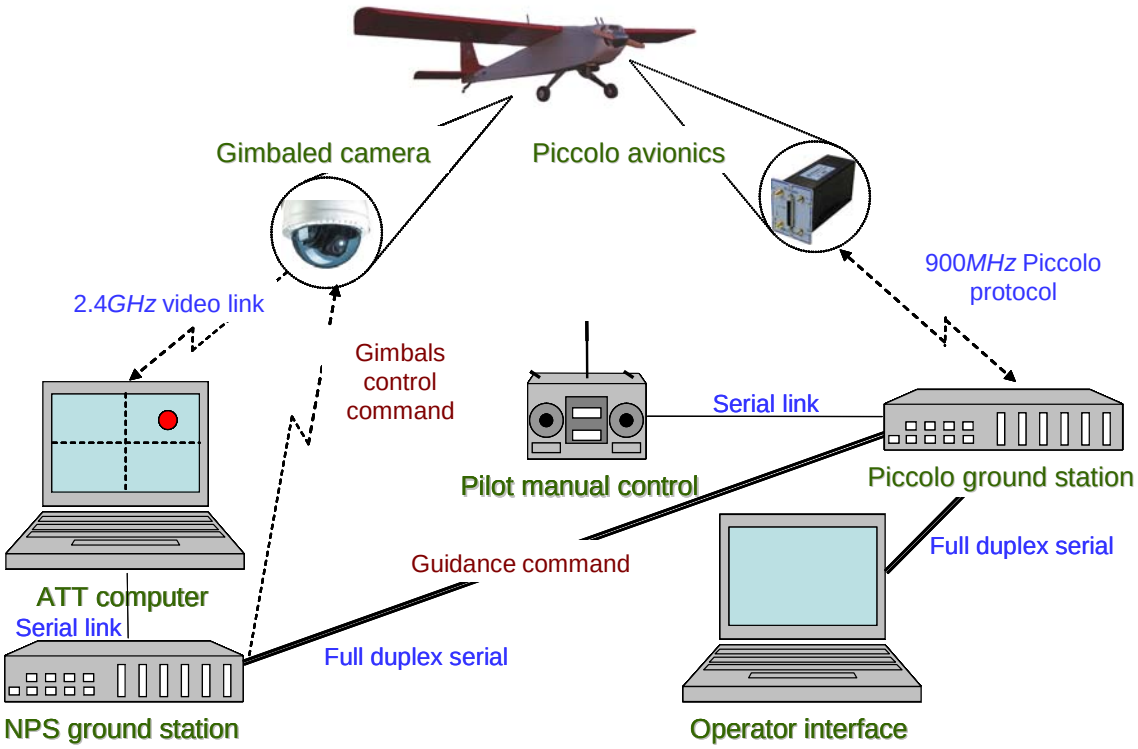

Figure 1. VBTT Architecture

B. SUAV

The SUAV is like the Predator UAV although it is a fraction of the size and price. Its small size, light weight, ruggedness and endurance provide a very useful tool on the battlefield.

Its primary payload is a gimbaled camera, Figure (2), that has an independent control system, manually or autonomously controllable from the ground.

Figure 2. Gimbaled Camera.

The SUAV itself, Figures (3) and (4), is a modified hobby aircraft called the Senior Telemaster. It comes in at 2.5 meters across the wings and at a weight of 8kg. The SUAV is powered by a 23 cm^3 gasoline powered two stroke engine that, when coupled with its 1500 cm^3 gas tank, has the ability to loiter for around three hours.

The SUAV carries the Piccolo avionics package which includes the INS; Piccolo allows the SUAV to be autonomous.

Figure 3. SUAV.

Figure 4. SUAV and Camera.

C. CONTROL SYSTEM

The SUAV is equipped with an onboard integrated GPS/INS system which also transmits position, velocity, acceleration, Euler rates and angles to the ground. Computers on the ground, using this supplied telemetry data can then use it to control the flight of the aircraft and make the SUAV navigation autonomous. Two types of autopilot control are used on the SUAV, The Piccolo Autopilot system and the NPS Autopilot. Both use the telemetry data which comes from the Piccolo avionics payload onboard the aircraft, Figure (5). Rate gyros, accelerometers, a GPS receiver and a pressure sensor are all part of the avionics payload. The payload communicates via a 900 MHz radio link to the ground station, Figure (6).

Figure 5. Piccolo Avionics Payload.

Figure 6. Current Ground Station.

1. Piccolo Autopilot

Piccolo was developed by Cloud Cap Technology. It has a user interface compatible with Windows run on a laptop computer in the field. The user interface displays a map of the local area and allows the user to manipulate waypoints while the SUAV is in flight and view telemetry data being sent from the aircraft. This data is saved in a file as it arrives.

The SUAV can be put into autonomous mode or be switched into manual mode at the press of a button. When in autonomous mode Piccolo controls the aircraft by flying the aircraft to waypoints. However, when a target is locked, NPS Autopilot is engaged and sends commands through the Piccolo software to the SUAV.

2. NPS Autopilot

NPS Autopilot was developed at the Naval Postgraduate School. It is an autopilot system used once a target is locked to fly the SUAV around the target around the trajectory that converges to a circle. Flying the SUAV around the target in a circle enables calculations to be made to estimate the range to the target, Equation(0.1).

$$\rho = \frac{v}{\dot{\lambda}} \tag{0.1}$$

Where $\dot{\lambda}$ is the angular rate of the LOS to the target, v is the velocity of the SUAV and ρ is the estimated range from the SUAV to the target. With the estimated range to the target, the location of the SUAV, the Euler angles of the aircraft as well as the gimbal angles all known, the target position can be estimated.

D. CAMERA

The camera, attached to a gimbal pan/tilt unit, is the primary payload of the SUAV. Video data is sent to the ground from the UAV through a 2.4 GHz omni-directional antenna and received through a high gain tracking antenna. Once received the video information is time stamped and displayed through PerceptiVU on a computer video screen. PerceptiVU is the program used to lock onto a target.

1. Gimbal Pan/Tilt Unit

While searching for a target the Gimbal Pan/Tilt Unit is under manual control from the ground. While in manual mode the user observes real-time video through a computer screen and moves the gimbal with a computer joystick. The range of the pan/tilt unit is 90 degrees tilt (strait ahead to directly at the ground) and 360 degrees pan. The movement of the joystick is converted into commands for the gimbal. Gimbal commands

are sent via one way communication to the SUAV through a 900 MHz radio link. The gimbal is under autonomous control when locked on a target.

2. PerceptiVU

PerceptiVU is image tracking software. Developed by PerveptiVU inc. this is the software we use to lock onto a target and stay locked on. A target lock is achieved by the user manning the joystick. The user initially moves the gimbal so that the target is in the PerceptiVU video screen and presses the trigger on the joystick. Once locked onto a target, information provided by PerceptiVU software is used to control the gimbal autonomously. While locked on, PerceptiVU outputs the location of the target in the camera frame. The output is given in pixels. The pixel where the target sits in the horizontal direction in the camera frame is called the “U” pixel. The vertical pixel is the “V” pixel. The PerceptiVU frame has 320 pixels horizontal and 240 pixels vertical resolution.

During the screen shot in Figure (7) PerceptiVU would output pixel values of +30V and +40U for example. The gimbal commands while locked onto a target are based on this U, V data. The pitch rate commands to the gimbal depend on the V pixel. The yaw rate commands to the gimbal depend on the U pixel. These commands are meant to keep the target in the center of the camera frame.

When the user wants to disengage and regain manual control of the gimbal the trigger is pulled again and the target is disengaged.

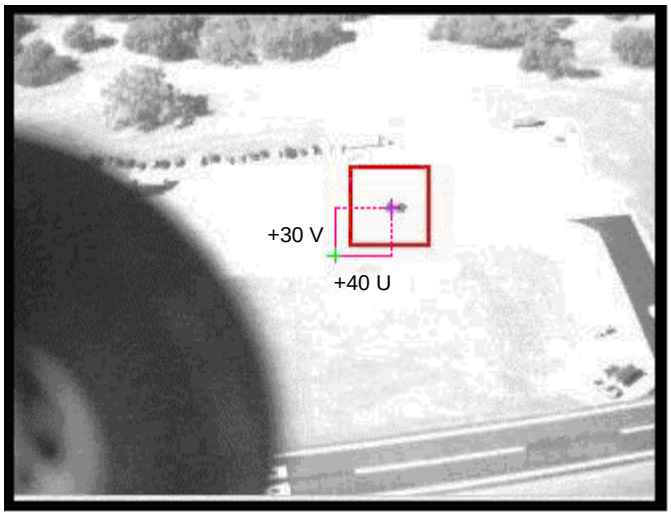

Figure 7. Screen Shot from PerceptiVU.

II. PROBLEM FORMULATION

A. PROBLEM

Errors in the target position estimation range on average from 20 to 50 meters. Some of these errors are attributed to imperfect measurements of Euler angles by Piccolo INS and of gimbal angles by NPS ground station. The purpose of this thesis is to use available flight test data to quantify the errors in the Gimbal angle measurement.

The approach adopted in this thesis is twofold:

- Using known target and UAV positions, use Euler and gimbal angle measurements to determine where the target would appear in the camera frame. Compare computed Target position with actual taken during the flight test
- Using PerceptiVU measurements, determine the gimbal angles that would result in these measurements. Compare these resultant angles with flight test data.

1. Gimbal Angle Errors

The gimbal data likely represents a source of errors. In this work it is assumed that commands sent to the gimbal are executed immediately, therefore output position of the camera attached to it is equal to the input command. However, extensive experimentation shows that this in not the case. Several factors distort this input-output relation:

- Calibration of the gimbal;
- Time delay introduced by wireless RF link;
- Noise due to the atmospheric and engine noise.

B. AVAILABLE DATA

To address these problems we will use the following flight test data:

- Position of the target in the camera frame provided by PerceptiVU;
- The location of the SUAV in geodetic coordinates;
- The location of the target in geodetic coordinates;
- The Euler angles of the aircraft in radians;
- The gimbal angles in radians;

- Video recording from the camera on the specific flight.

From this data we can calculate orientation of the camera. Using this orientation and the known location of the target we can estimate the location of the target in the camera frame and compare it to the true data taken from PerceptiVU. The difference in this location of the target in the camera frame will provide us insight into the errors in the gimbal angles of the SUAV. To calculate the estimated pixel location MATLAB 7.0 Simulink was used. A Simulink implementation of the approaches outlined in II.A is presented in Figure (8).

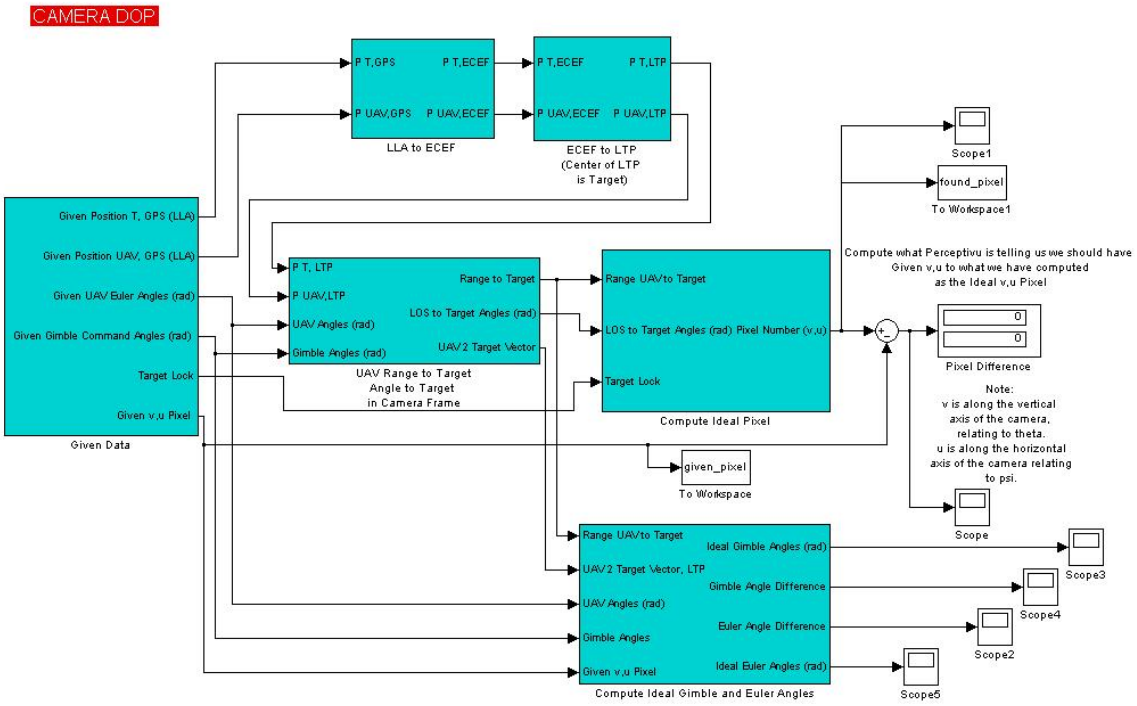

Figure 8. Entire Simulink Model

1. Loading Data

Loading the data is the first block of the Simulink diagram. For each flight of the SUAV its telemetry, gimbal and PerceptiVU data is saved versus real-time in the form of a MATLAB .mat file. The data loaded at the first step of the Simulink model, Figure (8) is presented in Table 1.

Data (units)	Subfile	Element
Target Position (rad,m)	N/A	N/A
SUAV Latitude (rad)	Telem	1
SUAV Longitude (rad)	Telem	2
SUAV Altitude, above ground (m)	Telem	3
SUAV Roll, psi, φ_G (rad)	Ctrl	24
SUAV Pitch, theta, θ_G (rad)	Ctrl	25
SUAV Yaw, psi, ψ_G (rad)	Ctrl	26
Gimbal Pitch, tilt, α_G	Gmbl	1
Gimbal Yaw, pan, β_G	Gmbl	2
Gimbal Lock	Gmbl	3
PerceptiVU U_{true} (pixel)	Gmbl	4
PerceptiVU V_{true} (pixel)	**Gmbl**	5

Table 1. **Data input into simulink diagram.**

The Simulink block that loads the data can be seen in Appendix (A.A).

2. Geodetic to ECEF

Flight test data provides the position of the target and SUAV in geodetic coordinates, Figure (9). In order to solve our task we need to transfer coordinates into a Local Tangent Plane (LTP). This transformation is done in two steps: first, is rotation from geodetic to ECEF (Figure (10)); second is to rotate from ECEF to LTP.

First step is accomplished through Equation(0.2).

$$\begin{aligned} r_\lambda &= \frac{r_e}{\sqrt{1-\varepsilon^2 \sin^2 \varphi}} \\ x^e &= (r_\lambda + h)\cos\varphi\cos\lambda \\ y^e &= (r_\lambda + h)\cos\varphi\sin\lambda \\ z^e &= ((1-\varepsilon^2)r_\lambda + h)\sin\varphi \end{aligned} \tag{0.2}$$

Where:

- φ is the Latitude angle (rad)
- λ is the Longitude angle (rad)
- h is the height above sea level (m)

Implementation of these equations using Simulink viewable in Appendix (A.B).

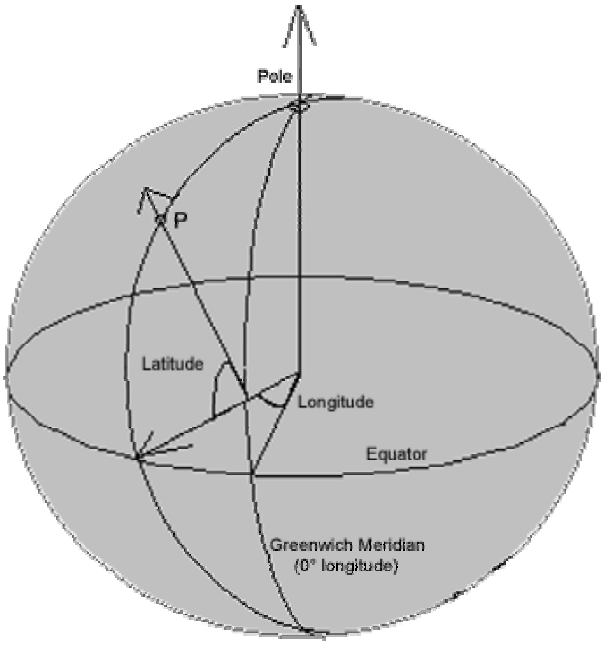

Figure 9. Geodetic Coordinate System.

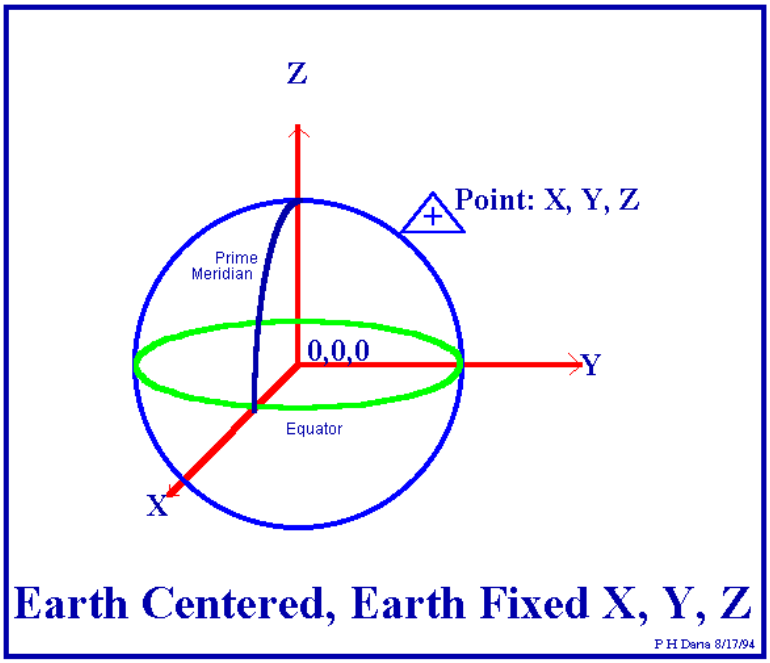

Figure 10. ECEF Coordinate System.

3. ECEF to LTP

LTP is a right hand rule reference frame which has its origin at a chosen spot. For this case the LTP origin is placed at the target position. The North-East-Down (NED) orientation of the LTP is assumed. Equation (0.3) shows the rotation matrix used to transform ECEF to LTP.

$$ {}^{NED}_{e}R = \begin{bmatrix} -\sin\varphi\cos\lambda & -\sin\varphi\sin\lambda & \cos\varphi \\ -\sin\lambda & \cos\lambda & 0 \\ -\cos\varphi\cos\lambda & -\cos\varphi\sin\lambda & -\sin\varphi \end{bmatrix} \tag{0.3} $$

φ and λ in Equation (0.3) are the Latitude and Longitude in the geodetic frame respectively.

$$ {}^{e}p_{SUAV} = {}^{NED}_{e}R * ({}^{e}p_{SUAV} - {}^{e}p_{target}) \tag{0.4} $$

The position of the SUAV in LTP coordinates, ${}^{NED}p_{SUAV}$ with components $[{}^{NED}x_{SUAV}, {}^{NED}y_{SUAV}, {}^{NED}z_{SUAV}]$, is found through Equation (0.4) where ${}^{e}p_{SUAV}$ is the position of the SUAV represented in ECEF coordinates, and ${}^{e}p_{Target}$ is the location of the target in ECEF coordinates. Equation (0.5) calculates the position of the target in LTP frame. Recall, ${}^{e}p_{Target}$ is the origin and should be equal to [0,0,0].

$$ {}^{NED}p_{Target} = {}^{NED}_{e}R * ({}^{e}p_{Target} - {}^{e}p_{target}) \tag{0.5}$$

Implementation of this transformation using Simulink can be seen in Appendix (A.C).

4. Target Position in Camera Frame

With the Target position and SUAV position in LTP frame we can calculate the range from the SUAV to the Target in Equation(0.6).

$$\rho = \sqrt{({}^{NED}x_{SUAV})^2 + ({}^{NED}y_{SUAV})^2 + ({}^{NED}z_{SUAV})^2} \tag{0.6}$$

Consider Figure (11) and Figure (12). Angular position of the target in the camera frame is given by ${}^{c}\theta_{LOS,T}$ and ${}^{c}\psi_{LOS,T}$.

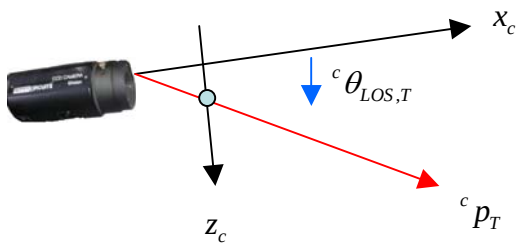

Figure 11. Visual Representation of ${}^c\theta_{LOS,T}$.

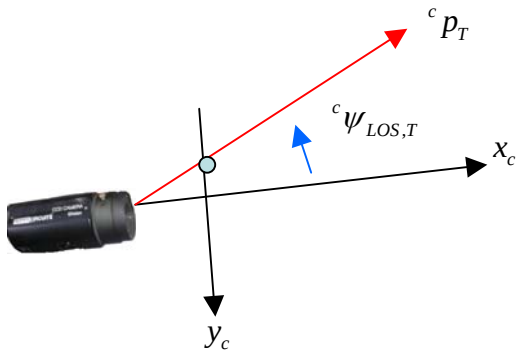

Figure 12. Visual representation of ${}^c\psi_{LOS,T}$.

These two angles are computed using the vector from the SUAV to the Target and two rotation matrices. Because we let the target position be the origin of the LTP frame, the vector from the SUAV to the Target is $-{}^{NED}p_{SUAV}$. The first rotation matrix, Equation(0.7), uses the Given Gimbal Angles, where α_G is the given tilt gimbal angle and β_G is the given gimbal pan angle, from flight test data. The second rotation matrix, Equation(0.8), uses the Given Euler Angles from Piccolo telemetry.

$$ {}^b_g R = \begin{bmatrix} \cos\alpha_G \cos\beta_G & -\sin\beta_G & -\sin\alpha_G \cos\beta_G \\ \cos\alpha_G \sin\beta_G & \cos\beta_G & -\sin\alpha_G \sin\beta_G \\ \sin\alpha_G & 0 & \cos\alpha_G \end{bmatrix} \tag{0.7} $$

$$
{}^{n}_{b}R = \begin{bmatrix} \cos\psi_G \cos\theta_G & \cos\psi_G \sin\theta_G \sin\varphi_G - \sin\psi_G \cos\varphi_G & \cos\psi_G \sin\theta_G \cos\varphi_G \\ \sin\psi_G \cos\theta_G & \sin\psi_G \sin\theta_G \sin\varphi_G + \cos\psi_G \cos\varphi_G & \sin\psi_G \sin\theta_G \cos\varphi_G - \cos\psi_G \sin\varphi_G \\ -\sin\theta_G & \cos\theta_G \sin\varphi_G & \cos\theta_G \cos\varphi_G \end{bmatrix}
$$

(0.8)

Equation(0.9) rotates the vector from the SUAV to the Target, $-{}^{NED}p_{SUAV}$, around the flight test data gimbal angles to give us the position of the target in the body frame, ${}^{b}p_T$.

$$
{}^{b}p_T = {}^{b}_{g}R * (-{}^{NED}p_{SUAV}) \tag{0.9}
$$

Equation(0.10) rotates the position of the target in the body frame to the position in the camera frame, ${}^{c}p_T$.

$$
{}^{c}p_T = {}^{n}_{b}R * {}^{b}p_T \tag{0.10}
$$

Position of the Target in the camera frame, ${}^{c}p_T$, has components $[{}^{c}x_T, {}^{c}y_T, {}^{c}z_T]$. These components are used in the computation of the tilt angle to the target in the camera frame, ${}^{c}\theta_{LOS,T}$, in Equation (0.11) and Figure (11). The pan angle to the target in the camera frame, ${}^{c}\psi_{LOS,T}$, is found in Equation (0.12) and Figure (12).

$$
{}^{c}\theta_{LOS,T} = a\tan\left(\frac{{}^{c}z_T}{{}^{c}x_T}\right) \tag{0.11}
$$

$$
{}^{c}\psi_{LOS,T} = a\tan\left(\frac{{}^{c}y_T}{{}^{c}x_T}\right) \tag{0.12}
$$

The angular location of the target in the camera frame and the range are used to calculate pixel number in the camera frame, Figure (13).

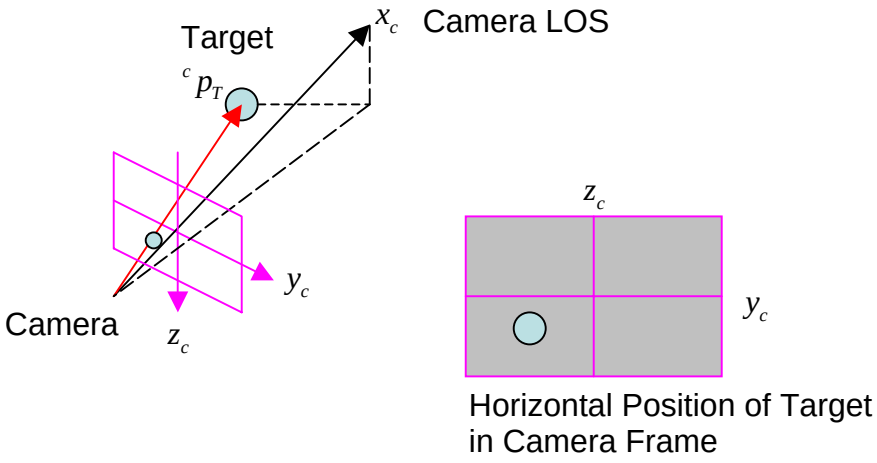

Figure 13. Target Projection onto the Camera Frame

Implementation of the equations in this section using Simulink can be seen in Appendix (A.D).

5. Pixel Calculation

Our first task calls for us to compare $V_{estimate}$ and $U_{estimate}$ with V_{true} and U_{true}. In order to compute $V_{estimate}$ and $U_{estimate}$ we need the following:

- Range from SUAV to Target, ρ
- Field of view of the camera, θ_{FOV} and ψ_{FOV}
- Angular position of the target in the camera frame, $^{c}\theta_{LOS,T}$ and $\psi_{LOS,T}$

To find θ_{FOV} and ψ_{FOV} we set up the camera and pointed it at a wall. We measured the distance from the camera frame to the wall, d. One the wall we measured the width, y, and height, z, of the camera's field of view.

$$\theta_{FOV} = 2 * a\tan\frac{z}{d} \tag{0.13}$$

$$\psi_{FOV} = 2 * a\tan\frac{y}{d} \tag{0.14}$$

Equation (0.13) shows us the field of view (FOV) in the vertical direction, θ_{FOV}. Equation (0.14) finds the horizontal FOV, ψ_{FOV}.

Once we know the camera field of view in the horizontal and vertical directions we can compute the pixel size, Equation (0.15) and (0.16) respectively. Note that 240 is the number of pixels in the PerceptiVU window vertically while there are 320 pixels across the screen horizontally.

$$v_{pixsz} = \frac{2\rho\tan(\frac{\theta_{FOV}}{2})}{240} \tag{0.15}$$

$$u_{pixsz} = \frac{2\rho\tan(\frac{\psi_{FOV}}{2})}{320} \tag{0.16}$$

The distance from the center of the camera frame to the target position in the camera frame, vertically and horizontally, is calculated in Equations (0.17) and (0.18) respectively.

$$v_{dist} = \rho\tan({}^{c}\theta_{LOS,T}) \tag{0.17}$$

$$u_{dist} = \rho\tan({}^{c}\psi_{LOS,T}) \tag{0.18}$$

This distance divided by pixel size, Equations (0.19) and (0.20), gives us the position of the target in the camera frame with pixels as the units.

$$V_{estimate} = \frac{v_{dist}}{v_{pixsz}} \tag{0.19}$$

$$U_{estimate} = \frac{u_{dist}}{u_{pixsz}} \tag{0.20}$$

Implementation of the equations in this section using Simulink can be seen in Appendix (A.E).

C. ANALYSIS OF THE RESULTS. CONCLUSION

The $V_{estimate}$ and $U_{estimate}$ pixels do not match up to V_{true} and U_{true}, Figure (14). This offset in the pixel location of the target in the camera frame proves to us that there is an error. This data suggests to us that there is a bias error in the pan angle of the camera and a larger bias in the tilt angle of the camera.

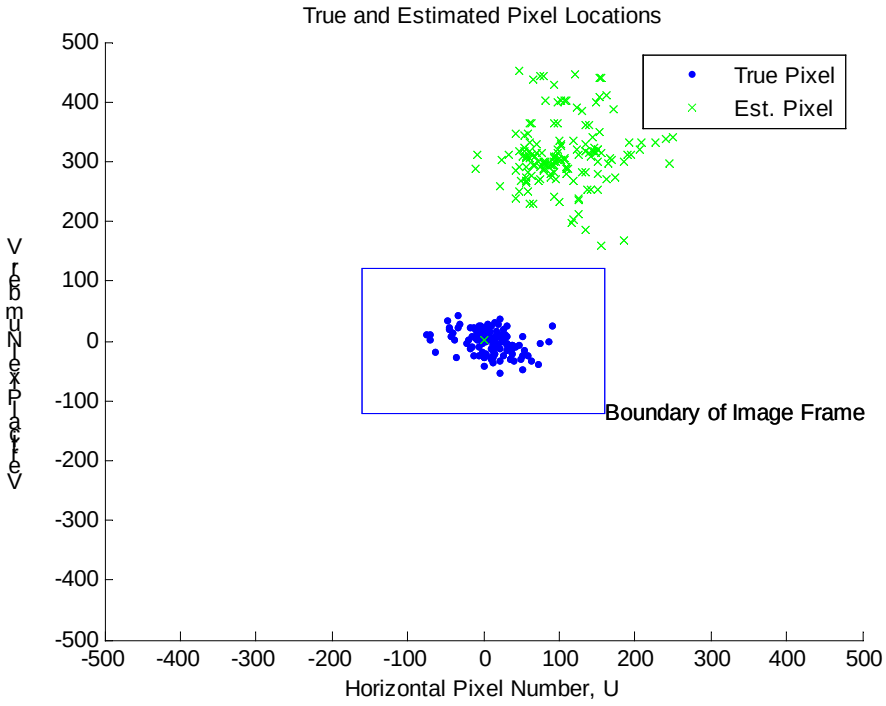

Figure 14. Given vs. Found Pixel Location of the Target in Camera Frame.

The mean V_{true} pixel number is -1.7041 and the mean U_{true} pixel number is 7.863 for this set of data. This is near the center of the camera frame. The Estimated pixels, however, are not near the center of the camera frame. The mean $V_{estimate}$ pixel number is 306.2 and the mean $U_{estimate}$ pixel number is 104.5. The difference between $V_{estimate}$ and V_{true} is 307.9 pixels. The difference between $U_{estimate}$ and U_{true} is 96.63 pixels. The mean location of the vertical pixel wouldn't even put the target into the visible camera frame since the visible frame only goes to +120 pixels vertically, Figure (15).

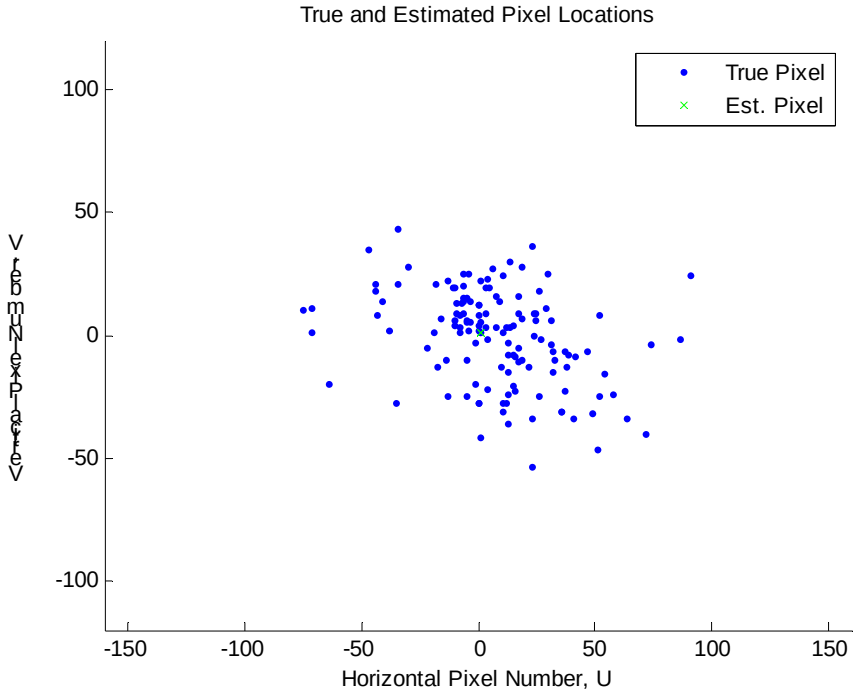

Figure 15. Plot of camera window.

The standard deviation of the True Pixels , $\sigma_{T\_pixel} = [19.42, 27.71]$, seen in Figure (16) is much lower than the standard deviation of the Estimated Pixels, $\sigma_{Est\_pixel} = [65.83, 48.37]$, seen in Figure (17).

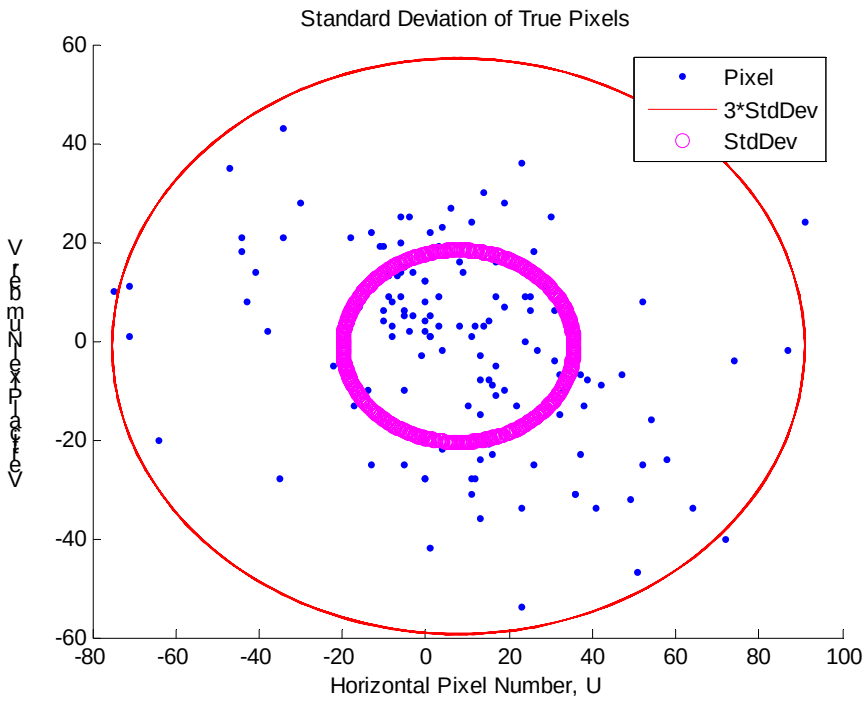

Figure 16. True Pixel Standard Deviation.

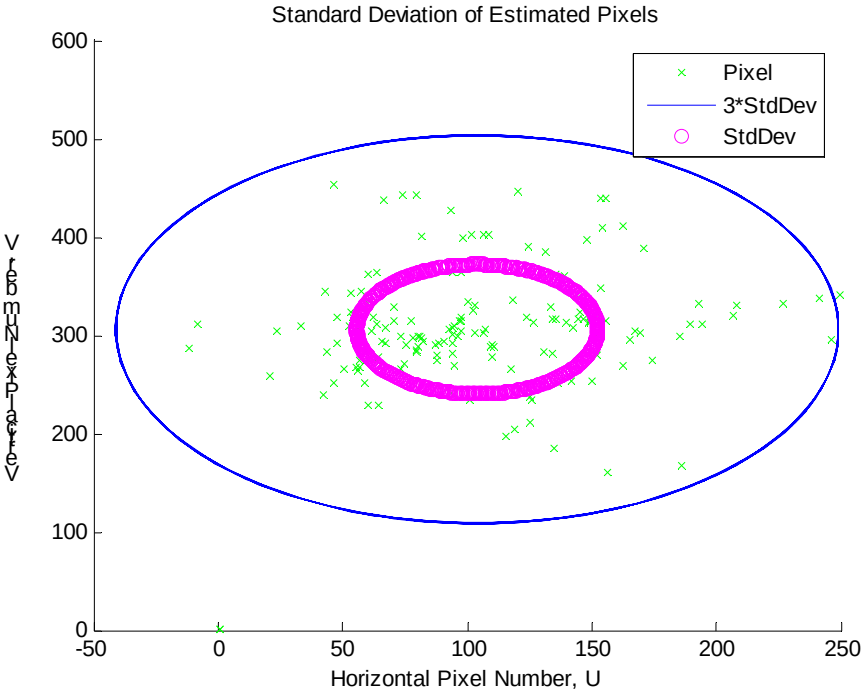

Figure 17. Estimated Pixel Standard Deviation.

In conclusion, the comparison between the $V_{estimate}/U_{estimate}$ pixels and the True pixels demonstrates that there is a biased constant error, that might be modeled as a fixed bias. The error comes from the Given Gimbal and Euler Angles. Using these angles for position estimation will therefore lead to a coherent error in the estimated target position.

III. SYSTEM DEVELOPMENT

A. OVERVIEW

Since there are errors in the orientation of the LOS we must find the source of this error. The LOS vector is used in the SUAVs most important mission, estimating the geodetic location of a target. With an incorrect LOS vector the NPS Autopilot commands will be off as well causing the SUAV to circle around a biased point, but not the target's position.

To find where the most errors come from in the system we must first calculate Gimbal angles that would match $V_{estimate}$ and $U_{estimate}$ pixel location of the target with V_{true} and U_{true} pixel location. If the two pixel locations match, we know the LOS from the UAV to the target is correct and a more accurate target position is being estimated. For every time step the Ideal Gimbal Angles (α_I and β_I) will be computed so that the estimated and true pixels equal to one another.

The "Ideal" gimbal angles will be computed and compared with the gimbal angles that we have from the flight test data, Given angles. The difference between the Given and Ideal Gimbal Angles will show us the error.

B. COMPUTING FOUND ANGLES

The Found Angles (α_F, β_F) are those angles that will direct the camera LOS to point directly toward the target. Transformation (0.21) is used to rotate a unity vector of the LOS in camera frame ($[1,0,0]$) to inertial frame ($[x_{LOS}; y_{LOS}; z_{LOS}]$) by using three consecutive rotations:

- ${}^{g}_{c}R$ - from Camera to Gimbal.
- ${}^{b}_{g}R$ - from Gimbal to Body.
- ${}^{n}_{b}R$ - from Body to Inertial.

$$\begin{bmatrix} x_{LOS} \\ y_{LOS} \\ z_{LOS} \end{bmatrix} = {}^{n}_{b}R_{g}^{b}R_{c}^{g}R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{0.21}$$

The rotation matrix ${}^{g}_{c}R$, which contains the angles found later in equations (0.26) and (0.27), is removed from this part of the process, since those angles are not yet know, and is re-introduced when solving for the Ideal Angles in part C of this section.

Found Gimbal Angles are computed while using the Given Euler Angles in the ${}^{n}_{b}R$ rotation matrix of Equation(0.22).

$${}^{b}_{n}R \begin{bmatrix} x_{LOS} \\ y_{LOS} \\ z_{LOS} \end{bmatrix} = {}^{b}_{g}R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{0.22}$$

Equation (0.23) through (0.25) calculate Found Gimbal Angles. Since the gimbal only uses two angles and we have three known values we can solve Equation(0.23).

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \cos\beta_F \cos\alpha_F \\ \sin\beta_F \cos\alpha_F \\ -\sin\alpha_F \end{bmatrix} \tag{0.23}$$

Simplified out we can solve for the Found α_F, Equation (0.24), and Found β_F, Equation (0.25).

$$\alpha_F = -a\sin(z) \tag{0.24}$$

$$\beta_F = a\tan(\frac{y}{x}) \tag{0.25}$$

The Found Angles for the gimbal, α_F and β_F, are shown in Figure (18) and Figure (19).

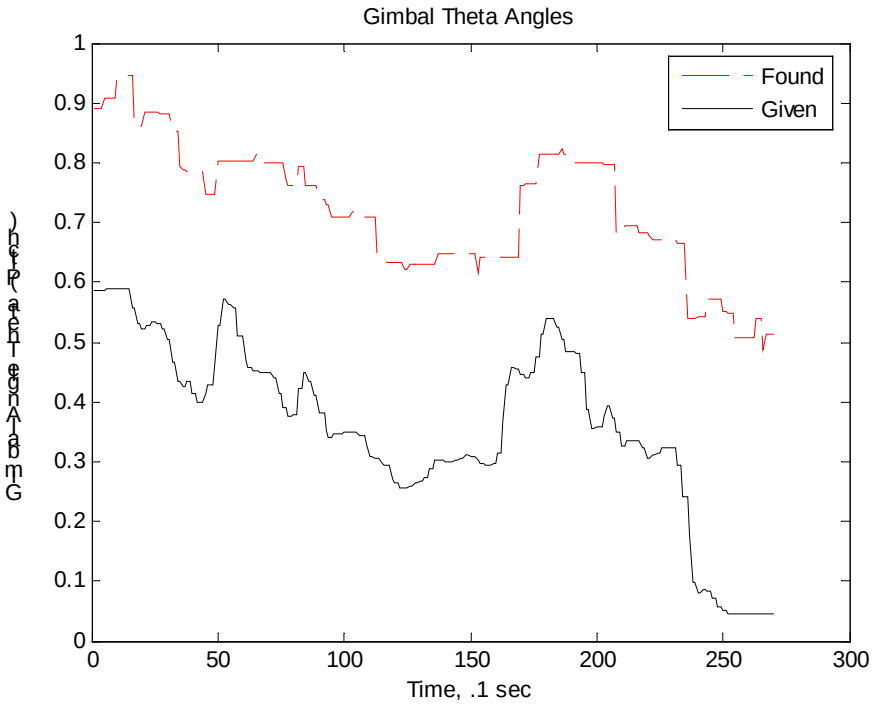

Figure 18. α_F **vs.** α_G**.**

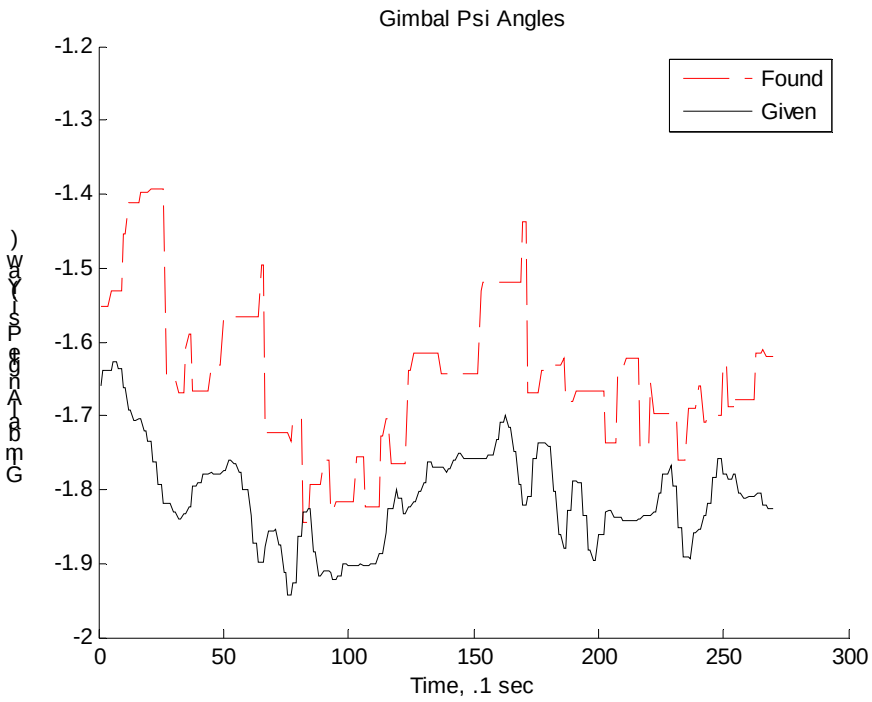

Figure 19. β_F **vs.** β_G**.**

Notice that in most of the Found Angles there is a very noticeable, almost constant, difference from the given data. Although the Found Angles are not the Ideal Angles, they still give us an initial idea of where the errors are occurring in the system. Gimbal α and β angles both appear to follow the same trend but with a bias between them. Computing the Ideal Angle will give us the final angles we are looking for and we can draw our conclusions from there.

Implementation of this section using Simulink is in Appendix (A.F).

C. COMPUTING IDEAL ANGLES

The Found Angle is calculated to give us a decent initial condition to feed into the iterative process used to calculate the Ideal Angles. The Found Angle points the camera directly at the target. Looking at our flight data, however, and the V_{true} and U_{true} pixels it gives us, we know that the camera is rarely pointed exactly toward the target, Figure (7). We know the camera is slightly off the direct line pointing toward the target and the target is moving around the screen. Computing the Ideal Angles takes into account this

small difference between the vector directly to the target and to the true center of the camera frame.

1. Simulink Portion

The angles between the true camera vector and the true pixel location can be calculated as follows using the calculated pixel size and range.

$$^{c}\theta_{LOS,T\_TRUE} = a\tan(\frac{V_{true} * v_{pixsz}}{\rho}) \tag{0.26}$$

$$^{c}\psi_{LOS,T\_TRUE} = a\tan(\frac{U_{true} * u_{pixsz}}{\rho}) \tag{0.27}$$

Once these true LOS to target angles are computed a MATLAB code is used to compute the Ideal Angles.

The data we have obtained from Simulink and will use to find the Ideal Angles are:

- Given Euler Angles (b_g.mat)
- Given and Found Gimbal Angles (g_g.mat and g_f.mat)
- SUAV to Target Vector, $-^{NED}p_{SUAV}$ (los.mat)
- True LOS to True Target Position Angles, $^{c}\theta_{LOS,T\_TRUE}$, $^{c}\psi_{LOS,T\_TRUE}$ (l.mat)

Parenthesis indicates the file name that this data is saved to after each run of the model.

2. MATLAB Portion

The data obtained in Simulink is saved and run through a MATLAB function to calculate the set of Ideal Angles. The .m file called gimbal.m, Appendix (B.B), is used to calculate the Ideal Gimbal Angles.

The FMINUNC command, Appendix (B.A,C), run in MATLAB calls upon the gimbal.m file. The FMINUNC function iterates through combinations of angles until the condition of the FMINUNC command is satisfied, Equation(0.28). When the condition of FMINUNC is satisfied, the LOS vector and the vector [1;0;0] rotated through the three rotation matrices are equal.

$$F = norm(\begin{bmatrix} x_{LOS} \\ y_{LOS} \\ z_{LOS} \end{bmatrix} - {}^{b}_{n}R\, R^{g}_{b} R^{c}_{g} R \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}) \tag{0.28}$$

When searching for the Ideal Gimbal Angles, the angles inside of the ${}^{g}_{b}R$ rotation matrix are the ones that will be changed until the condition is satisfied. The angles inside ${}^{b}_{n}R$ are the Given Euler Angles from Piccolo. The angles inside ${}^{c}_{g}R$ were calculated with Equations (0.26) and (0.27).

Once the FMINUNC function's condition is satisfied it will output the Ideal Angles. As was said earlier the Found Angles are used as initial conditions or the starting point in finding the Ideal Angles. It was decided that the Found angles were satisfactory as an initial condition, because the actual position of the target is never too far away from the center of the camera frame, where the Found Angles are pointing.

Figure (20) is included to show visually how the FMINUNC function converges at its solution. At a specific time in the data set the FMINUNC function runs through gimbal angles until the minimum of the function is reached.

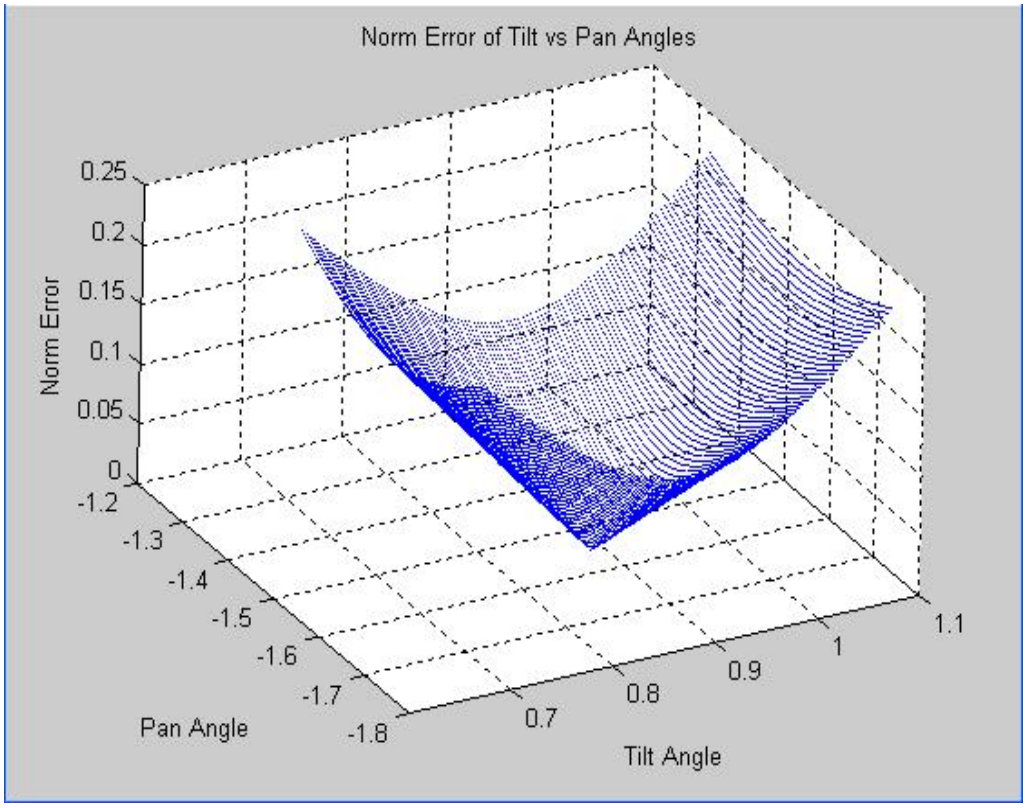

Figure 20. FMINUNC Process Wide View

Looking closer towards the minimum of Figure (20) we get Figure (21). Readings from the figure gives us a tilt (α_I) value at the minimum found between .885 and .89, near .8855. The pan (β_I) value at the minimum is between -1.552 and -1.554, near -1.5535. The FMINUNC function would then output [.8855,-1.5535] as the answer at this specific time. These are the Ideal Angle we are looking for.

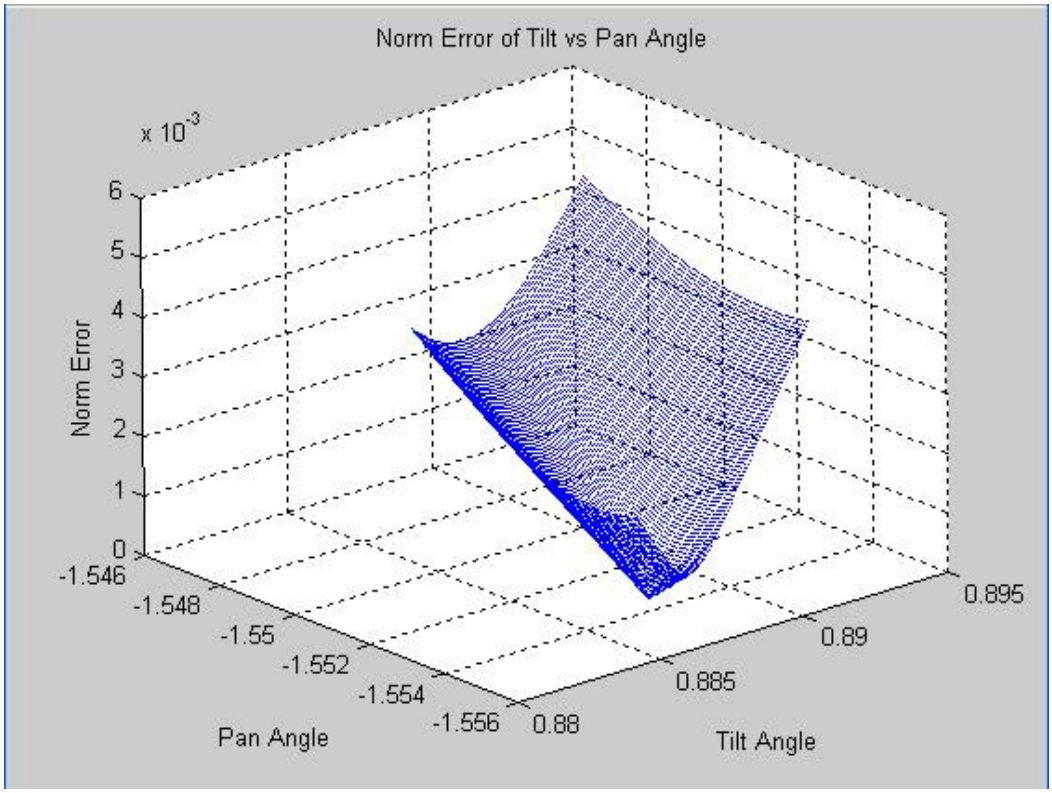

Figure 21. FMINUNC Process Zoom.

IV. RESULTS

A. IDEAL ANGLES

The FMINUNC function gave us the results illustrated in Figures (22) and (23).

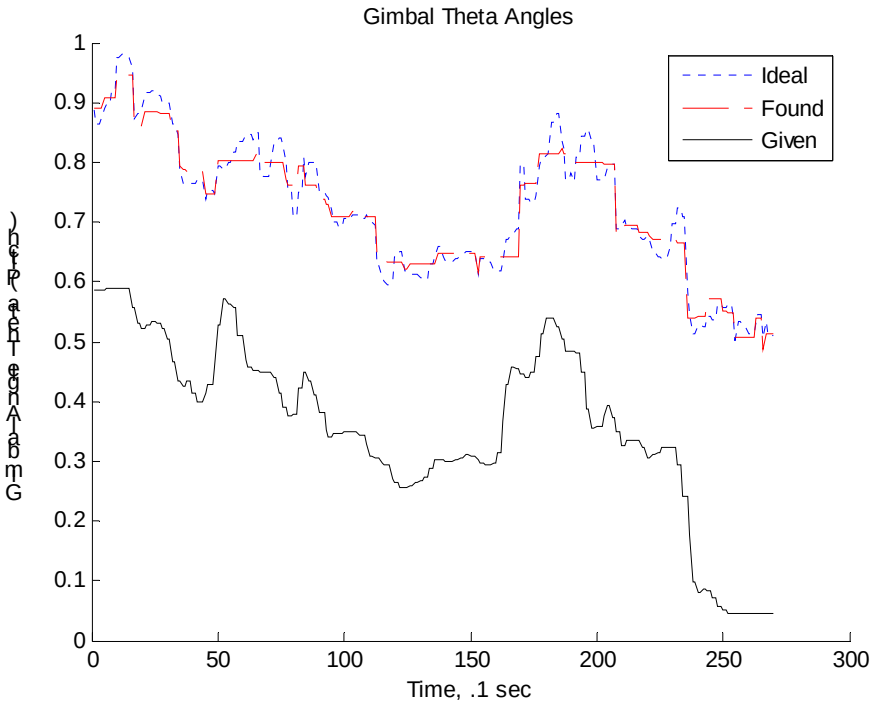

Figure 22. α_I **vs.** α_F **vs.** α_G**.**

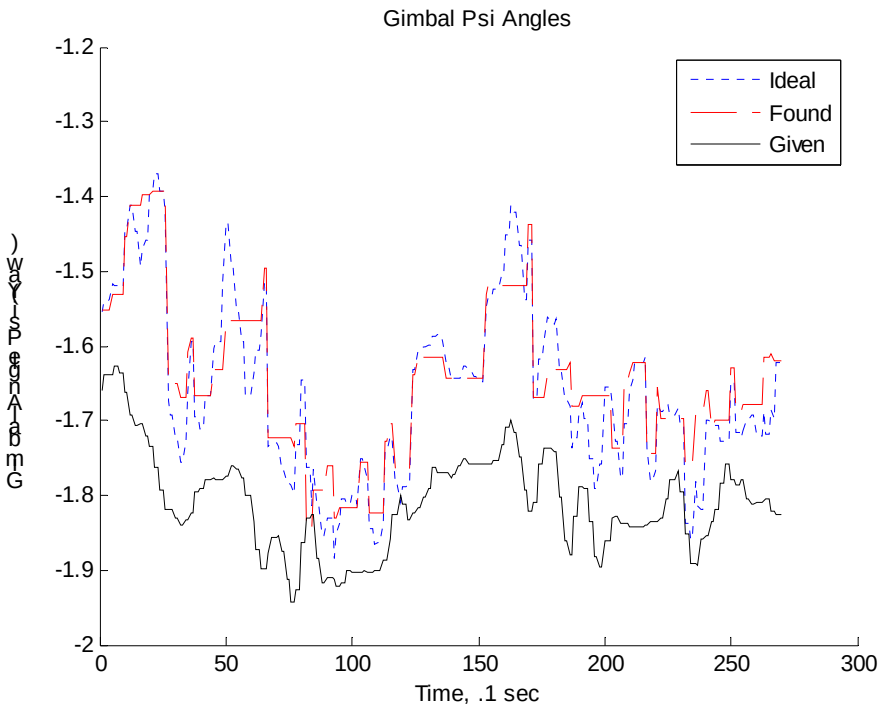

Figure 23. β_I **vs.** β_F **vs.** β_G**.**

B. COMPARING RESULTS

1. Gimbal Angles

Upon visual inspection it follows that the Given Gimbal Angles follow the same path as the Ideal Gimbal Angles but are offset from one another by a certain bias, Figure (22) and Figure (23). The offset or bias are plotted in Figure (24) and Figure (25). This could be attributed to errors in the calculation of the Ideal Gimbal Angles or it could be caused by some bias error in the given data.

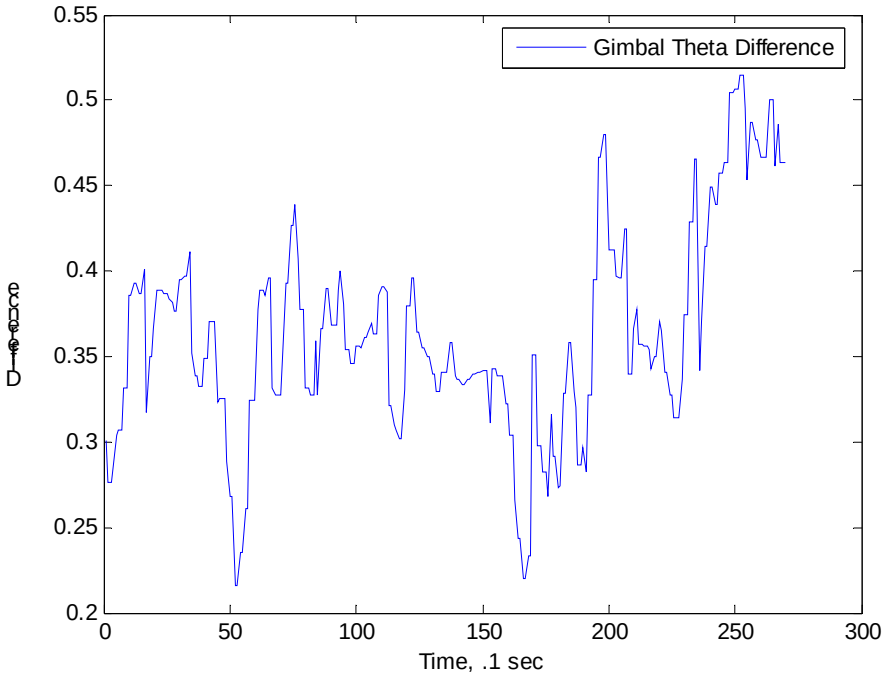

Figure 24. Difference Between Ideal α_I and Given α_G.

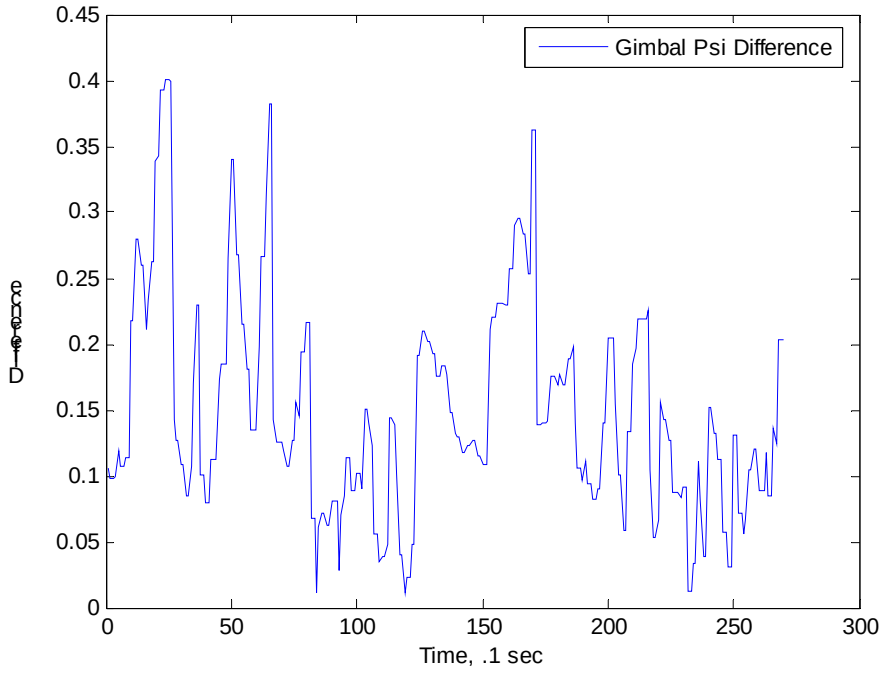

Figure 25. Difference Between Ideal β_I and Given β_G.

It suggests that a simple addition or gain in the gimbal command could fix the bias in the α and β angles. Fixing the bias would match the Given Angles to the Ideal Angles. Eliminating the bias in the Gimbal Angles from the NPS ground station readiness/telemetry would reduce the error in the target position estimation.

C. RECOMMENDATION

My recommendation to the project is to invest additional resources into the following projects.

- A model of the gimbal that will account errors induced while sending data to the gimbal, i.e. time delay, gimbal movement delays.
- Taking the bias between the Ideal Gimbal Angles and the Given Gimbal Anlges into account when using the Given Gimbal Angle for position estimation.
- In depth Gimbal calibration investigation.
- A more accurate measuring device onboard the SUAV for better readings of its Euler Angles.

It appears that some changes can be made in the gimbal loop that will not raise the price of the SUAV. In the future, research should be done on modeling the gimbal dynamics and looking deeper into the bias found there. If that does not solve the problem, then a better avionics package onboard the SUAV might solve our problems.

APPENDIX A: SIMULINK BLOCK DIAGRAMS

A. LOAD DATA BLOCK

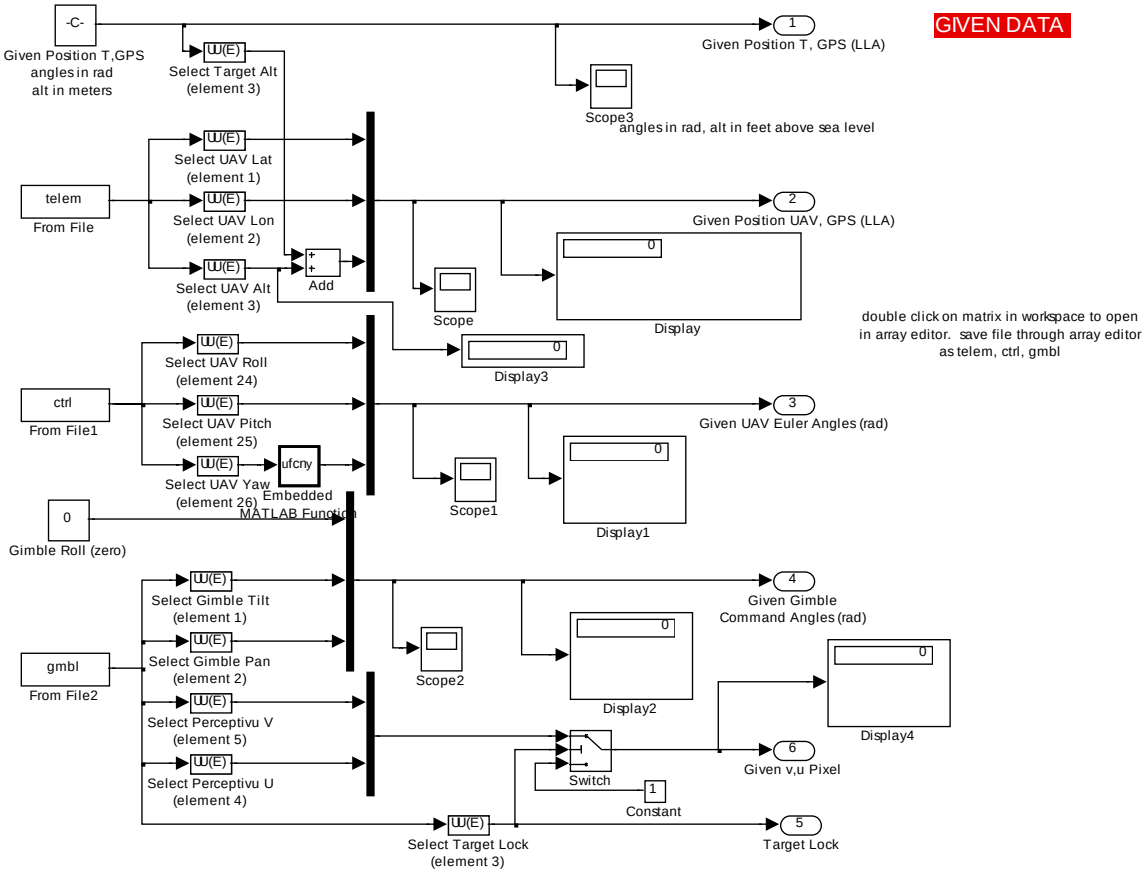

Figure 26. Load Data Block.

B. LLA TO ECEF BLOCK

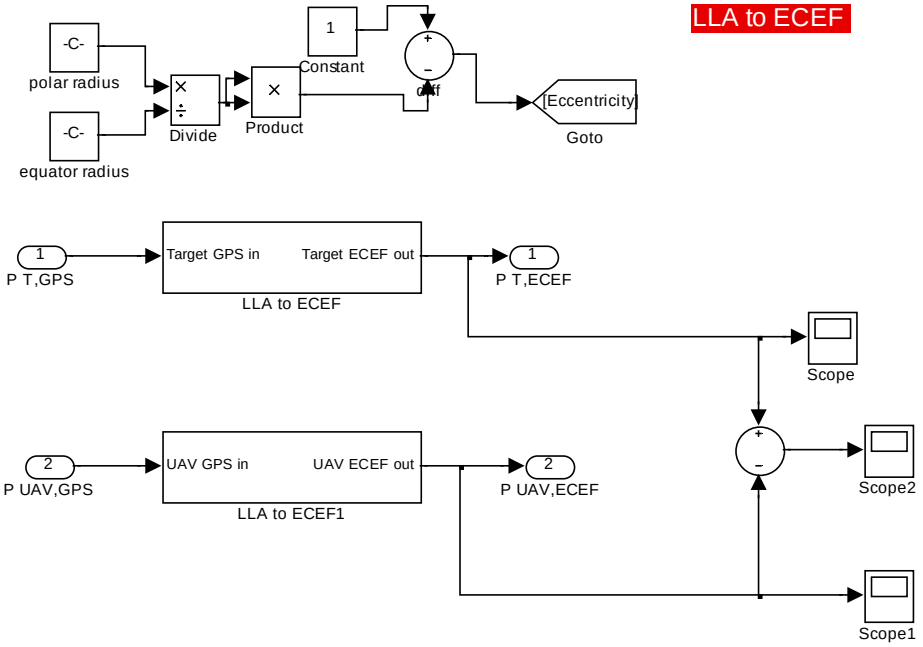

Figure 27. LLA to ECEF Block.

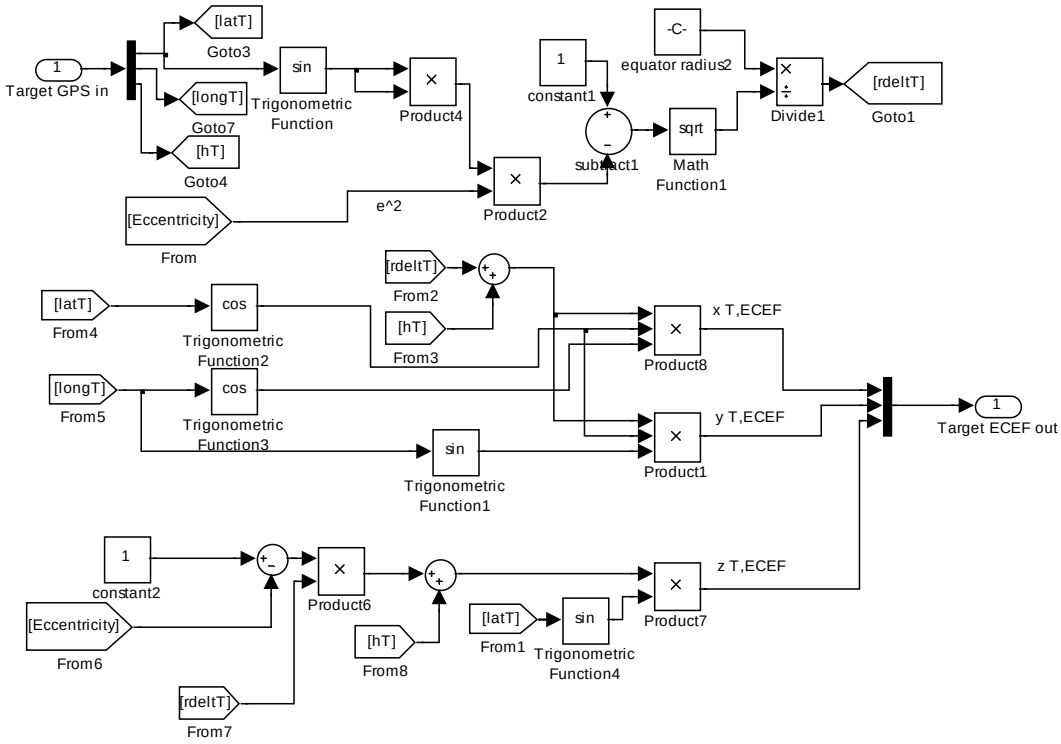

Figure 28. Implementation of ECEF Equations in LLA to ECEF Block.

C. ECEF TO LTP BLOCK

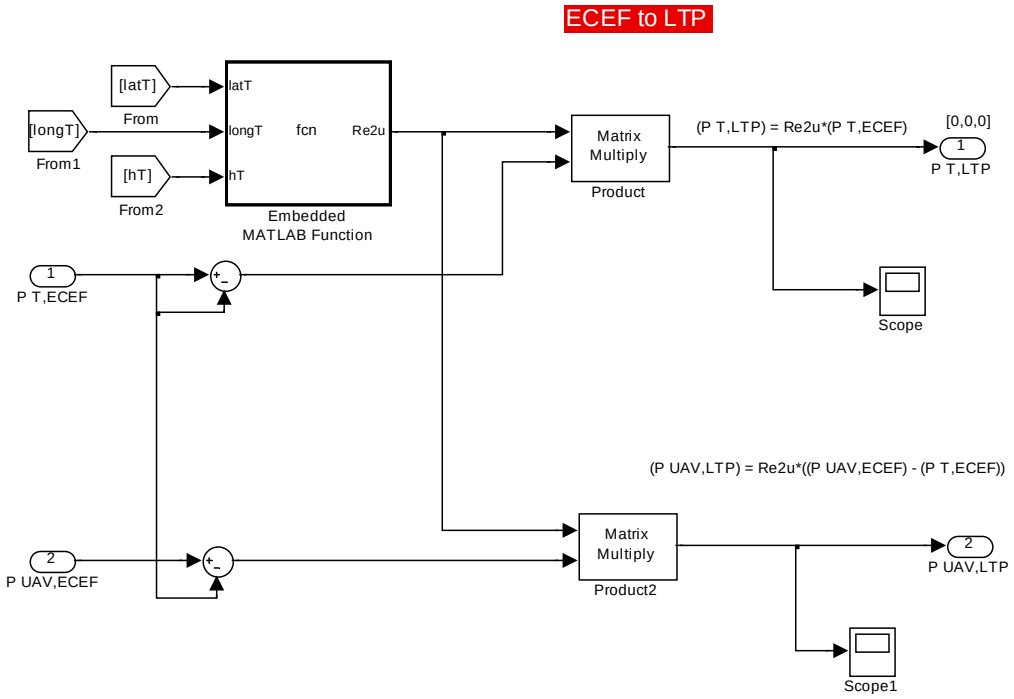

Figure 29. ECEF to LTP Block.

The MATLAB code within the Embedded MATLAB Function of Figure (39).

```
function Re2u = fcn(latT,longT,hT)
% This block supports an embeddable subset of the MATLAB language.
% See the help menu for details.


Re2u=[cos(latT)*cos(longT) cos(latT)*sin(longT) sin(latT);
    -sin(longT) cos(longT) 0;
    -sin(latT)*cos(longT) -sin(latT)*sin(longT) cos(latT)];

RNED=[0 0 1;0 1 0;-1 0 0];

Re2u=RNED*Re2u;
```

D. TARGET POSITION IN CAMERA FRAME BLOCK

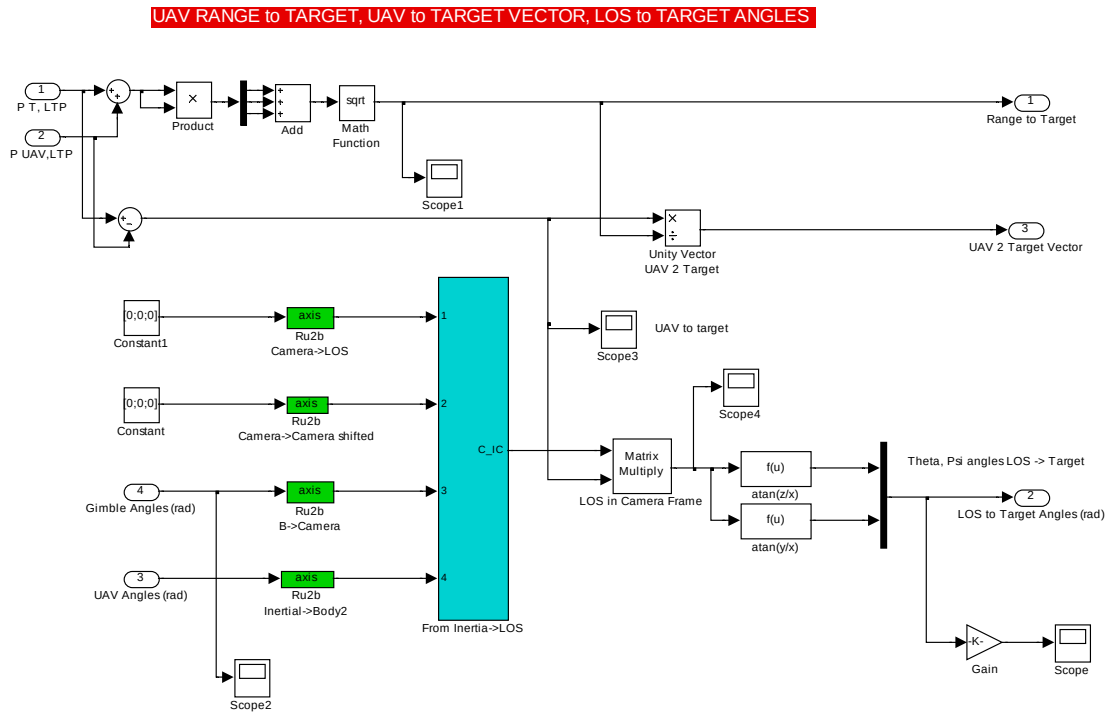

Figure 30. Target Position in Camera Frame Block.

E. COMPUTE PIXEL BLOCK

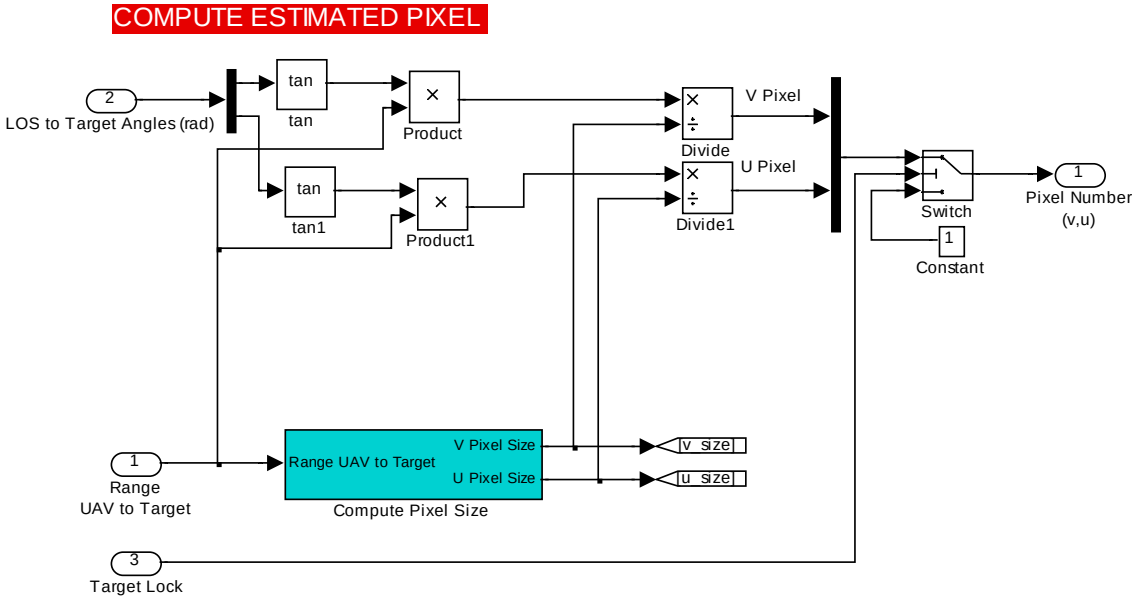

Figure 31. Compute Estimated Pixel Block.

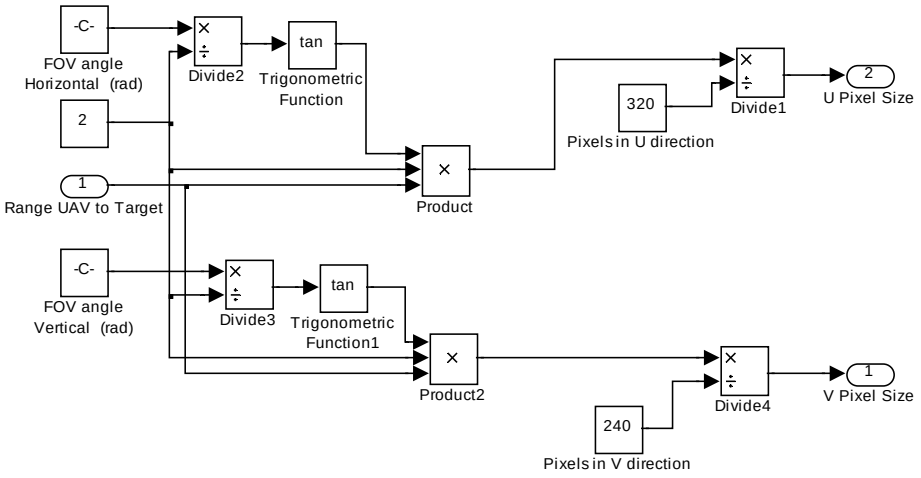

Figure 32. Compute Pixel Size Block.

F. COMPUTE FOUND ANGLES BLOCK

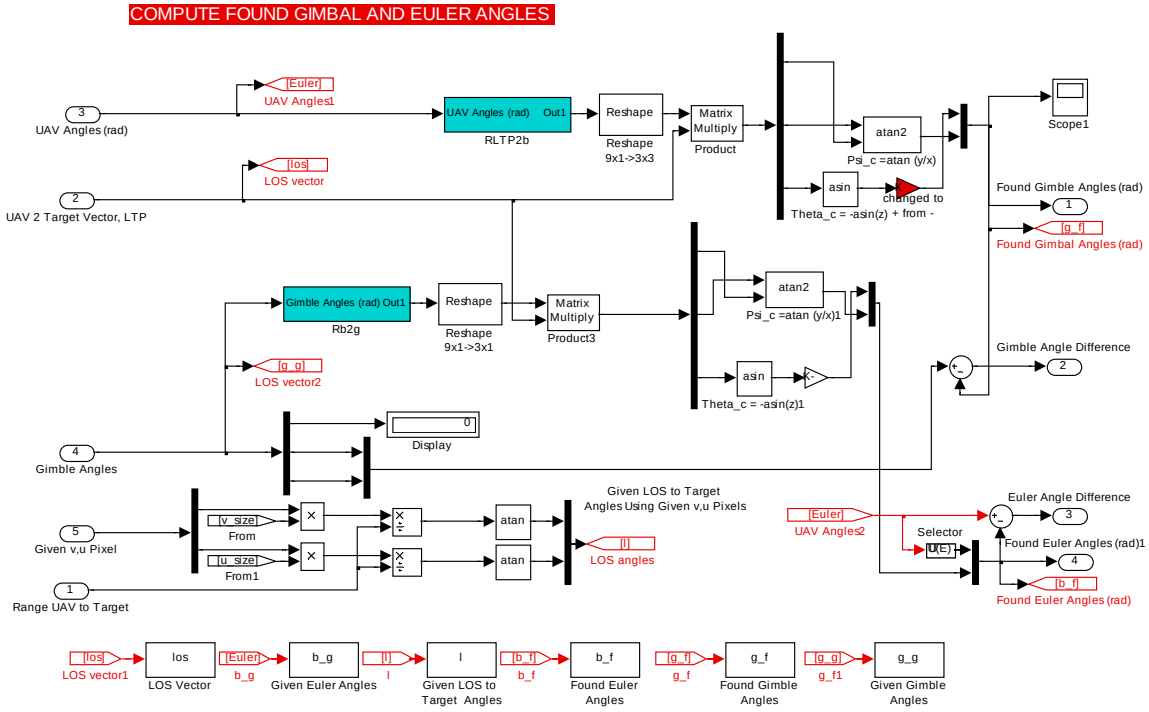

Figure 33. Compute Found Angles Block.

THIS PAGE LEFT INTENTIONALLY BLANK

APPENDIX B: MATLAB CODE

A. FMINUNC MATLAB HELP FILE

```
FMINUNC finds the minimum of a function of several variables.
    X=FMINUNC(FUN,X0) starts at X0 and attempts to find a local minimizer X
    of the function FUN. FUN accepts input X and returns a scalar function
    value F evaluated at X. X0 can be a scalar, vector or matrix.

    X=FMINUNC(FUN,X0,OPTIONS) minimizes with the default optimization
    parameters replaced by values in the structure OPTIONS, an argument
    created with the OPTIMSET function.  See OPTIMSET for details.  Used
    options are Display, TolX, TolFun, DerivativeCheck, Diagnostics,
    FunValCheck GradObj, HessPattern, Hessian, HessMult, HessUpdate,
    InitialHessType, InitialHessMatrix, MaxFunEvals, MaxIter,
    DiffMinChange and DiffMaxChange, LargeScale, MaxPCGIter,
    PrecondBandWidth, TolPCG, TypicalX. Use the GradObj option to specify
    that FUN also returns a second output argument G that is the partial
    derivatives of the function df/dX, at the point X. Use the Hessian
    option to specify that FUN also returns a third output argument H that
    is the 2nd partial derivatives of the function (the Hessian) at the
    point X. The Hessian is only used by the large-scale method, not the
    line-search method.

    [X,FVAL]=FMINUNC(FUN,X0,...) returns the value of the objective
    function FUN at the solution X.

    [X,FVAL,EXITFLAG]=FMINUNC(FUN,X0,...) returns an EXITFLAG that describes
    the exit condition of FMINUNC. Possible values of EXITFLAG and the
    corresponding exit conditions are

      1  FMINUNC converged to a solution X.
      2  Change in X smaller than the specified tolerance.
      3  Change in the objective function value smaller than the specified
          tolerance (only occurs in the large-scale method).
      0  Maximum number of function evaluations or iterations reached.
     -1  Algorithm terminated by the output function.
     -2  Line search cannot sufficiently decrease the objective function along
          the current search direction (only occurs in the medium-scale method).

    [X,FVAL,EXITFLAG,OUTPUT]=FMINUNC(FUN,X0,...) returns a structure OUTPUT
    with the number of iterations taken in OUTPUT.iterations, the number
    of function evaluations in OUTPUT.funcCount, the algorithm used in
    OUTPUT.algorithm, the number of CG iterations (if used) in
    OUTPUT.cgiterations, the first-order optimality (if used) in
    OUTPUT.firstorderopt, and the exit message in OUTPUT.message.

    [X,FVAL,EXITFLAG,OUTPUT,GRAD]=FMINUNC(FUN,X0,...) returns the value
    of the gradient of FUN at the solution X.

    [X,FVAL,EXITFLAG,OUTPUT,GRAD,HESSIAN]=FMINUNC(FUN,X0,...) returns the
    value of the Hessian of the objective function FUN at the solution X.

    Examples
      FUN can be specified using @:
         X = fminunc(@myfun,2)

    where MYFUN is a MATLAB function such as:

        function F = myfun(x)
        F = sin(x) + 3;

      To minimize this function with the gradient provided, modify
      the MYFUN so the gradient is the second output argument:
         function [f,g]= myfun(x)
          f = sin(x) + 3;
          g = cos(x);
      and indicate the gradient value is available by creating an options
```

```
  structure with OPTIONS.GradObj set to 'on' (using OPTIMSET):
     options = optimset('GradObj','on');
     x = fminunc('myfun',4,options);

  FUN can also be an anonymous function:
     x = fminunc(@(x) 5*x(1)^2 + x(2)^2,[5;1])

If FUN is parameterized, you can use anonymous functions to capture the
problem-
dependent parameters. Suppose you want to minimize the objective given in the
function MYFUN, which is parameterized by its second argument A. Here MYFUN is
an M-file function such as

  function [f,g] = myfun(x,a)

  f = a*x(1)^2 + 2*x(1)*x(2) + x(2)^2; % function
  g = [2*a*x(1) + 2*x(2)               % gradient
       2*x(1) + 2*x(2)];

To optimize for a specific value of A, first assign the value to A. Then
create a one-argument anonymous function that captures that value of A
and calls MYFUN with two arguments. Finally, pass this anonymous function
to FMINUNC:

  a = 3;                             % define parameter first
  options = optimset('GradObj','on'); % indicate gradient is provided
  x = fminunc(@(x) myfun(x,a),[1;1],options)


See also optimset, fminsearch, fminbnd, fmincon, @, inline.
```

B. GIMBAL.M

```
function [F] = gimbal(x)
% this block sets up the function to be used in the optimization
%finding the best tilt and pan angles to show gimble command error

load los;
load b_g;
load l;
load g_f;
global i

LOS_v = [los(2,i);los(3,i);los(4,i)];

Rb2LTP=[cos(b_g(4,i))*cos(b_g(3,i))          sin(b_g(4,i))*cos(b_g(3,i))          -
sin(b_g(3,i));cos(b_g(4,i))*sin(b_g(3,i))*sin(b_g(2,i))-
(sin(b_g(4,i))*cos(b_g(2,i)))
sin(b_g(4,i))*sin(b_g(3,i))*sin(b_g(2,i))+(cos(b_g(4,i))*cos(b_g(2,i)))
cos(b_g(3,i))*sin(b_g(2,i));cos(b_g(4,i))*sin(b_g(3,i))*cos(b_g(2,i))+(sin(b_g(4,i
))*sin(b_g(2,i)))                        sin(b_g(4,i))*sin(b_g(3,i))*cos(b_g(2,i))-
(cos(b_g(4,i))*sin(b_g(2,i))) cos(b_g(3,i))*cos(b_g(2,i))].';

Rc2b=[cos(x(1))*cos(x(2))  cos(x(1))*sin(x(2))  sin(x(1));-sin(x(2))  cos(x(2))  0;-
sin(x(1))*cos(x(2)) -sin(x(1))*sin(x(2)) cos(x(1))].';

RLOS2c=[cos(l(2,i))*cos(l(3,i))  cos(l(2,i))*sin(l(3,i))  sin(l(2,i));-sin(l(3,i))
cos(l(3,i)) 0;-sin(l(2,i))*cos(l(3,i)) -sin(l(2,i))*sin(l(3,i)) cos(l(2,i))].';

F=norm(LOS_v-(Rb2LTP*Rc2b*RLOS2c*[1;0;0]));
```

C. GIMBAL_OPTIMIZATION.M

```
load g_f
options = optimset('TolX',.000001);
s=size(g_f,2);
i=1;
global i

for i=1:s
start=[g_f(2,i),g_f(3,i)];
[x,fval,exitflag] = fminunc(@(x)gimbal(x),start,options);
g_i(1,i)=x(1,1);
g_i(2,i)=x(1,2);
fval_g(i)=fval;
i=i+1
end
```

THIS PAGE LEFT INTENTIONALLY BLANK

LIST OF REFERENCES

1. Prince, Robert. *Autonomous Visual Tracking of Stationary Targets Using Small Unmanned Aerial Vehicles*. Master's Thesis, Naval Postgraduate School, Monterey, CA, 2004.

2. Kaminer, Isaac. *Introduction to Aircraft Navigation*. Lecture Course. Naval Postgraduate School, Monterey, CA, 2005

3. *Piccolo*. 18 Sept 2005 http://www.cloudcaptech.com/piccolo.htm

4. *Naval Postgraduate School Support for Combatant Commanders and the Office of the Secretary of Defense*. 21 Sept 05 <http://www.nps.edu/AboutNPS/NPSDistinctive/NPS%20COCOM-OSD%20Support.pdf>

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. Dr. Isaac Kaminer
   Naval Postgraduate School
   Monterey, California

4. Dr. Vladimir Dobrokhodov
   Naval Postgraduate School
   Monterey, California

5. Dr. Anthony Healey
   Chairman, Department of Mechanical Engineering
   Naval Postgraduate School
   Monterey, California